Newsletter for the Toronto
Local Chapter of the ACM
APL Special Interest Group
ISSN 1195-9878
Volume 2 Number 4
January 1995

Gimme Arrays!

Next meeting

Date Monday, Jan. 23
Time 6:30 pm
Location 16th Floor , BCE Place
William M. Mercer Ltd.
(Note: 6 floors above usual meeting place)

Address BCE Place
Canada Trust Tower
161 Bay Street

Topic *Surfing the Net — Vector style*
(see page 2)

Speaker Scott Parkinson
Hutchison Avenue Software Corporation

Free admission. All welcome!

Future meetings

Monday, February 27
Soliton
Monday, March 27
Meeting with an actuarial theme
Monday April 24 (t.b.a.)

In this issue

Surfing the Net	2
Featuring Soliton	2
Around the World: Northern California news	3
Apologies	5
APL95 Dates for your diary	5
APL95NET	7
Dyalog APL/W Version 7.1	8
A pleasure to read	10
Gilman & Rose of the 1990s?	11
Calendar of events	12

Due to space constraints in this issue, a number of items have been held over to February's **Gimme Arrays!**

Namespaces are impressive

Robert Byers

On October 24, 1994, **Peter Donnelly**, Managing Director of **Dyadic Systems Limited**, gave a ninety-minute presentation on Dyalog APL/W Version 7.0 in front of an audience of approximately sixty interested Toronto-based APLers. Version 7.0 is Dyadic's latest release, a substantial upgrade to its previous versions. It began shipping in August 1994.

Dyadic was the first APL vendor to introduce a true Microsoft Windows, 32-bit, APL development environment, beginning in 1992, with the introduction of version 6.2. That first release introduced a suite of high-level Windows development tools (quad functions) that interfaced easily with the Windows APL. In 1993, version 6.3 appeared and extended the

*(**Namespaces** continued on page 6)*

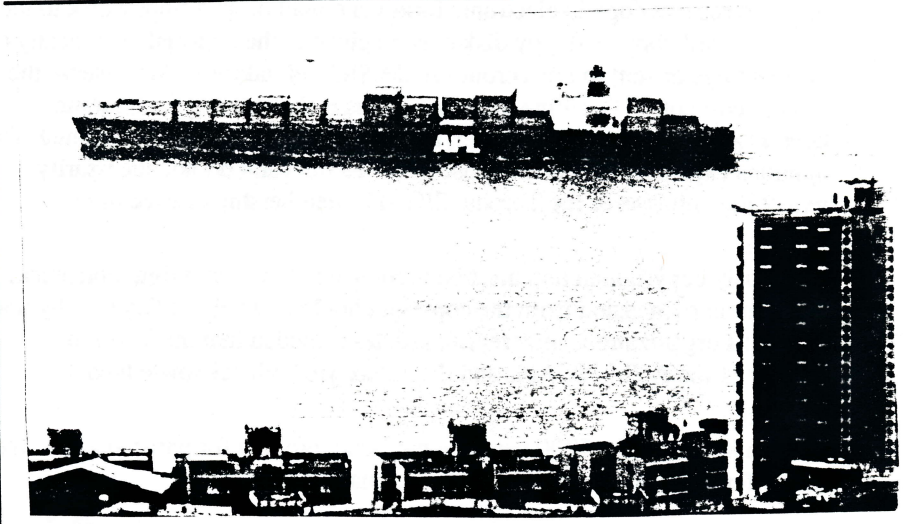
The Net result is the Best result

Diane Whitehouse

Did you ever see a roomful of APL enthusiasts acting like a neural net? Visually, you observe their bowed heads and scribbling pens. But most of all, it is their loud, collaborative discussions that hit you. Interestingly, APLers don't go in for the kind of competitive learning that **Ben Best** was aiming to illustrate. They share their knowledge and expertise actively. What they do looks more like simultaneous forward feeding and backward propagation. The conclusion is that APLers' brains are not Von Neumann machines. While there is not only a lot of parallel processing going on, they also have a wonderful tolerance for noise.

This highly interactive evening illustrated to the **Toronto APLSIG** how neural nets are designed and computed, and how sample

*(**Neural Nets** continued on page 4)*



APL going places in 1995.

Surfing the Net —Vector Style

Scott Parkinson will be giving the **Toronto APLSIG** an introduction to the Internet, what it is, how it works, the services available, how to get hooked up, its cost, and what it holds for the future.

He will log on to the Net for a surfing session showing some of the services on it and how they can be used. (He will target specific places and services on the Net of interest to APLers and J people.) The session will be fun and interactive, and open to questions at all times.

The meeting will be held at **William M. Mercer Limited** on the sixteenth floor at 161 Bay Street (six floors above the CIBC offices where the group usually meets).

Immediately following the meeting, APL Executive members, **Cameron Linton** and **Richard Procter** will initiate a discussion on Toronto's possibilities for setting up a World-Wide Web site.

__Scott Parkinson__ is Vice President of Finance with the __Hutchison Avenue Software Corporation__ based in Montreal. He is also the Technical Subcommittee Director for the __CIA Internet Task Force__, a task force established to bring services to actuaries using the Internet. Scott is a former APL user and a new J devotee.

Featuring Soliton

The February 27 meeting will provide an update on **Soliton Associates** since it became independent from Reuters two years ago. There will be a review of the current state of Sharp APL, both the traditional mainframe system and the newer Unix version. Finally, techniques for communicating among mainframe and Unix platforms, and for communicating among APL and non-APL environments will be described.

The presentation will be made by **Clarke Bruce** (President), **Nancy Lamb** (Manager of Sales and Marketing), and **Larry Moore** (Account Executive).

Gimme Arrays!

Submitting Articles

The editorial staff encourages submission of articles for publication. We prefer to receive copy in electronic form via e-mail or on a floppy disk to the editors. Hard copy or floppy disk may be given to the editorial team at any of the meetings or sent to the Toronto ACM SIGAPL address. We reserve the right to edit copy. Deadline for submissions is the meeting each month. ***Gimme Arrays!*** is your newsletter, and your contributions are welcomed. The opinions expressed herein are those of the authors and do not necessarily reflect the opinions of the Toronto SIGAPL membership or executive.

Copyright

Previously copyrighted text must be accompanied by a copyright notice and publication permission from the copyright holder. Other publications by not-for-profit organizations may reprint articles provided that the Toronto chapter of the ACM SIGAPL and the author are both acknowledged.

Advertising

Advertising rates are $200 per page per issue, pro rata for part-page ads. We also offer multiple-issue rates. Please contact the editor.

This newsletter has been printed on 100% recycled paper.

Toronto Special Interest Group on APL

Meetings

Start at 6:30 and are generally held on the fourth Monday of the month, from September to November and January to May.

Location

161 Bay Street
BCE Place (Canada Trust Tower)
10th Floor Cafeteria (CIBC/Wood Gundy)
Bay Street and Front Street
Toronto, Ontario

Address correspondence to

Toronto ACM SIGAPL
P.O. Box 384
Adelaide Street Post Office
Toronto, Ont. M5C 2J5

Annual Membership Fees

Ordinary $25; Student $5

Corporate Sponsorship

Contact the executive about promotion possibilities and meeting venues.

Chair

Cameron Linton
Phone: (416) 364-5361
crl@ipsalab.tor.soliton.com

Secretary & Newsletter Production

Marc Griffiths
Phone: (416) 532-0843
marcg@utcc.utoronto.ca

Newsletter Editor

Diane Whitehouse
Phone: (416) 539-0393
dianew@utcc.utoronto.ca

Treasurer & Membership

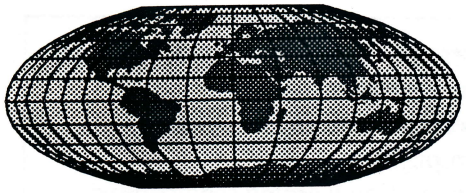
Eric Granz
Phone: (416) 980-7149
Fax: (416) 861-3791
egranz@ipsalab.tor.soliton.com

Information Co-ordinator

Richard Procter
Phone: (416) 651-4037
rjprocter@ipsalab.tor.soliton.com

Special Projects

Chris Burke
Phone (416) 369-1630
chris.burke@rose.com



Around the world: Northern California aims for outreach and companionship in a hardy vehicle

George Mendonsa

APL Bay Area User's Group is starting the new year with outreach, growth, APL95 and a name for the newsletter—*The Buggy*. This affectionate name has been most notably regularly referred to by **Curtis Jones** (our Secretary/Treasurer). So, the front cover depicts this trial as a name. Unless there are any objections, this is the newsletter's name. And now we need a logo.

Aside from the obvious allusion to the name of our group, I suppose *The Buggy* could be stretched as a symbol of outreach. A stylized picture of a buggy superimposed on a map of the bay area might well be an appropriate logo and capture the image of outreach and companionship in a hardy vehicle.

As far as future directions, we need to emphasize both local and worldwide goals. We need to get directly involved with local schools in presenting APL. The mathematics area offers a potentially fertile arena. To reinforce this, our SIG will be at the Mathematical Conference here in San Francisco from January 4–5, 1995.

One direction now of vendor emphasis is along the lines of a better "Visual Basic" development platform. However, what might provide a broader appeal would be user orientation instead or along with the developer's bent we see among all the vendors. The present direction seems to be potentially very limited.

APL Bay Area Users' Group's possible future user emphases include:

- Emphasis on the user—not the developer
- APL symbolic Math and APL algebras
- Extension of the language to generalize objects
- Compilers
- Outreach
  - Beginners' books and applications
  - High schools and colleges
  - Graphics and charting as an integral part of APL
  - Publicity for the wide diversity of APL applications
  - A commercial APL magazine—a cross of:
    - *Vector*
    - *Dr. Dobbs Journal*
    - *Circuit Cellar*
    - *Nibble*

*With his resignation as Editor of the APL BUG Newsletter, "The Buggy," **George Mendonsa** welcomes in the new year with this overview of potential new directions and endeavours. George can be contacted at men@crl.com.*

*Gimme Arrays! gratefully acknowledges the source of this editor's column by **George Mendonsa** as The Buggy, Dec. 1994-January 1995 issue.*

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆

☆ Torontonians might like to ☆
☆ think about their own arrays ☆
☆ wishlist for 1995. Could we get ☆
☆ involved in the **International** ☆
☆ **Math Olympiad** to be held at ☆
☆ York University in Summer ☆
☆ 1995 (see Calendar), for ☆
☆ example? Or, how can we ☆
☆ pursue the idea of a Toronto ☆
☆ **World-Wide Web site** (see ☆
☆ page 2). ☆

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆

Happy new year!. Season's greetings to the Toronto SIGAPL from Alexander Skomorokhov.

(Neural Nets continued from page 1)
networks operate. Ben began with an historical tour of the development of neural networks, and started with the work of McCulloch and Pitts in 1943. Many researchers thought a death knell had been sounded to neural nets in the late 1960s—when Marvin Minsky and his graduate student, Seymour Papert, proved that Perceptrons cannot be used for non-linear problems.

But neural net applications have proliferated since 1986 with the publicity given to the backpropagation algorithm by Rumelhart and McClelland. Today, eighty percent of neural network applications use backpropagation. The US Defense Department has been the leading funder of Artificial Intelligence research, and has recently been investigating the use of neural networks for missile guidance systems because of their "graceful degradation" when damaged. Ben also briefly outlined the use of neural nets by major credit card companies, with cervical Pap smear tests, and time-series analysis.

"Ben's professional interest in neural nets lies in the financial domain."

Backpropagation is a form of supervised learning. Its neurodes may also have a sigmoid function. Graduated descent minimizes any error. Used largely for non-linear problems, it forms the basis for three-quarters of all today's neural nets. Research is being undertaken to develop models that can recognize handwritten characters and human speech.

Attractor nets settle on certain patterns (like patterns of energy).

Most neural network applications currently in use can be placed in one of three major categories:

- Feed-forward networks using supervised learning (including Perceptrons, Adalines and Backpropagation)
- Feed-forward networks using unsupervised learning (competitive learning networks, e.g., Kohonen networks)
- Attractor networks (fully connected, recurrent networks, e.g., Hopfield Networks, Associative Memories and Boltzman Machines).

They generally involve many neurodes and are very sparse. Statisticians will be pleased to learn that attractor nets can solve the travelling sales representative problem.

Local University of Toronto professor, **Geoffrey Hinton**, is a world expert in the field of backpropagation. When one member of the audience exclaimed, "But Hinton doesn't use APL," another retorted on his behalf, "Yes, but imagine if he did!"

Ben's principal examples, on which he got the APL community in the audience to work, involved competitive learning. Operating on a "winner takes all" basis, in competitive learning each neurode is trained to recognize zeros and ones. With this kind of model, it is possible to vary the learning rates of the neurodes, their initial weights, and the number of times they iterate (see Figure 1).

Ben's professional interest in neural nets lies in the financial domain. There is now software package after package of neural nets available. But, as Ben says of most bond traders, "If you find an effective system, you're going to keep it to yourselves." If you want to read more about neural nets with a financial angle or in relation to arrays-related languages, Ben especially recommends reading work by J enthusiast, **Alvin Surkan**.

The core of all neural network algorithms is the manipulation of matrices of connection weights. Computationally, most of the CPU in neural network applications is spent on the inner product operation. Neural

(Neural Nets - continued on page 5)

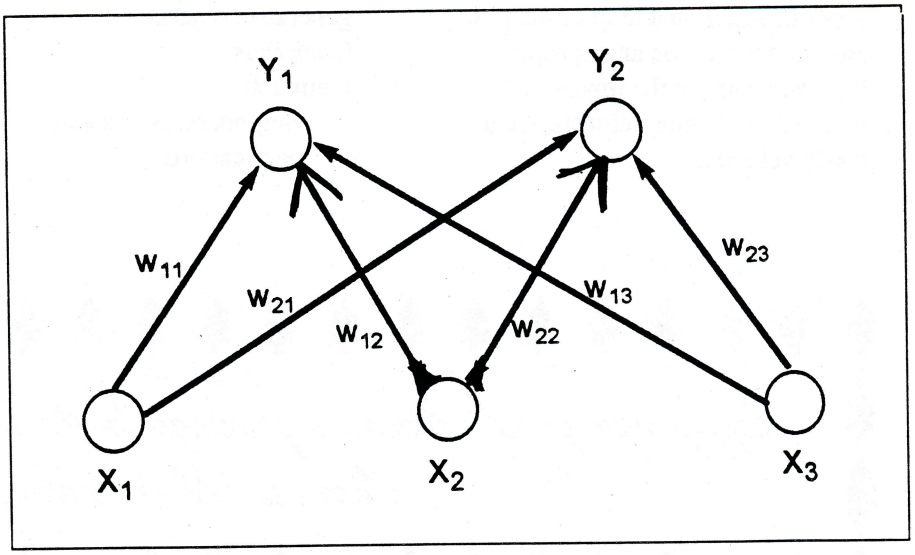


Figure 1. Competitive Learning Network

*(**Neural Nets** continued from page 4)*
network programming is intensely mathematical, which is why Mathematica is the language of choice for many neural network programmers. APL and J are also good for neural network programming for these reasons—and because their interactive character makes them suitable for the kind of "tweaking and fiddling" that neural network design often requires. The newer versions of APL and J that include control structures should help speed up learning algorithms requiring iteration (by 25%, says **Bob Bernecky**). Parallel machines, or vector-processing hardware that was specifically designed for APL and J, would give APL and J a massive advantage—especially for non-iterative algorithms—for both programming and execution-time.

We hope Ben strikes lucky with a neural net. We certainly wish him well in his endeavours to direct his initially hobbyist interest—in artificial intelligence and how the brain works—towards a career based even more firmly in the area of neural nets.

*Ben Best has been studying Neuroscience intensively while working as a Senior Programmer /Analyst on financial APL applications at **Scotia McLeod, Inc.** He is at ben.best@canrem.com.*

*Forty members of the APL community in Toronto were present at Ben's talk. Following the meeting, a dozen or so went for supper at the **Mövenpick Marché**.*

References

Hinton, G.E. (1992) How neural networks learn from experience. *Scientific American*. September Issue, 144-151.

McCulloch, W.S. and Pitts. W. (1943) A Logical Calculus of the Ideas Immanent in Nervous Activity in *The Bulletin of Mathematical Biophysics*, 5 (4), December, 115–133.

Minsky, M. 1968) (Editor) *Semantic Information Processing*. Cambridge, Massachusetts: MIT Press.

Rumelhart, D.E., McClelland, J.L. and the PDP research group (1986) *Parallel distributed processing: Explorations in the microstructure of cognition*. Vols I and II. MIT Press: Cambridge, Massachusetts.

Skurikhin, A.N. & A.J. Surkan (1994) A Parallel Correlation-Based Algorithm in J learns Neural Network Connections. In APL and its Applications. International Conference, 11-15 September 1994, Antwerp, Belgium, Conference Proceedings, edited by A. Delmotte, *APL Quote Quad*, 25 (1), September, 194–199.

Surkan, A. and J. Singleton (1993) Neural networks for bond rating improved by multiple hidden layers in *Neural networks in finance and investing. Using artificial intelligence to improve real-world performance*, edited by R.R. Trippi and Turban, E. Chicago, Illinois, pp275-287.

Apologies

In the last issue of **Gimme Arrays!**, we used the term "workspaces" in the article on **Tony O'Hagan** and **John Scholes'** work when we actually meant "namespaces." **Chris Burke** also noted that the first example of linear programming in Section 4 (page S2) of **Richard Brown**'s J supplement should read as follows:

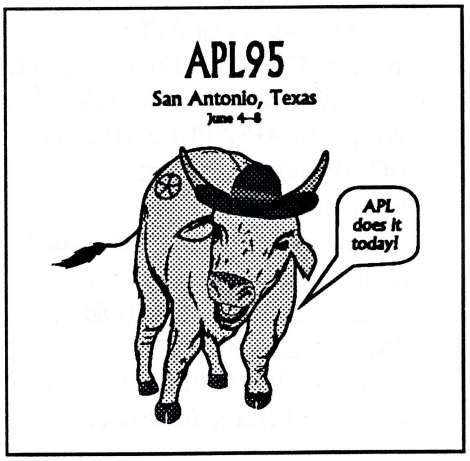
maximize $z = 2x + 3y$ subject to $4x + y \quad 12$ and $2x + 5y \quad 15$ with $x \quad 0$ and $y \quad 0$.

APL95 dates for your diary

If you missed the deadline for paper submissions to APL95 but would still like to be involved, here are some dates for your diary. The Invitation and Preliminary Program will be distributed soon. Act now!

April 5	Send in software
April 7	Last early bird registrations
May 1	Send in poster submissions
May 26	Deadline for SIG enclosures (newsletters, pamphlets) in conference package

Send your ideas and materials to APL95, PO Box 210367, Bedford, TX 76095, USA. Other points of contact include APL95_info@acm.org and APL95_conf@acm.org.



(*Namespaces - continued from page 1*)
graphical user (GUI) interface by adding more Windows objects, properties and events. Also introduced was a new workspace (WDESIGN) that gave APL/W a Visual Basic look-and-feel to building GUI event-driven applications. APL/W also has an excellent visual debugger which allows for easy step-through of functions and sub-functions and for the display, in separate windows, of local and global variables as they change or are built up.

"Workspaces have a tendency to grow—at least mine do....Namespaces allow you to set up a workspace in a much more organized manner."

Dyalog's main competitor, **Manugistics' APL*PLUS**, took a slower route in developing a true windows interface. Versions 4, 5 and 5.1 had some Windows extensions. But trying to develop any serious Window applications proved to be difficult and tedious. Dyalog APL/W clearly had a better product which was unchallenged until the introduction of APL*PLUS III, version 1.02. In my opinion, however, new features in version 7.0 re-establish Dyadic's leading technological edge.

All releases of APL/W work in either DOS/Windows 3.1, 3.11 (386 enhanced mode) or OS/2 2.1, Warp 3.0. APL*PLUS III runs in DOS/Windows 3.1 and OS/2 3.0 Warp only.

Major new features included in this latest release are namespaces, MDI (multiple document interface), VBX controls, ODBC support, a new grid object (for spreadsheet control), toolbar, statusbar, hint & tip objects, metafile support and a new session object that is customizable by the user. Even if you do not plan to develop Windows applications you will enjoy working with this product.

Peter did not spend much time talking about each of the new features. Instead, his talk focused on the most innovative new feature specific to APL/W called *namespaces*. In their simplest form, namespaces allow APL users to subdivide a workspace into separate and distinct parts. Peter described namespaces like using DOS subdirectories. Traditionally, APL workspaces are like a flat file: a workspace contains all your functions and variables. Workspaces have a tendency to grow—at least mine do. One of the problems with a large workspace is that you must ensure each function and variable name is unique.

Namespaces allow you set up a workspace in a much more organized manner. Name conflicts are not an issue because each namespace can contain APL objects with the same name. This is comparable to separate DOS sub-directories where you can have files of the same name. Peter demonstrated a number of examples, using namespaces, including running functions in other functions by using compound names. You reference functions and variables in other namespaces by including their namespace name separated by a period.

All GUI objects (i.e., forms, buttons etc.) in APL/W are automatically namespaces. APL functions and variables can, for example, "reside" inside a form (GUI container object). This extends the object-oriented programming principle (OOP) of "encapsulation" to APL. You can copy a GUI object from a workspace into another workspace and all associated objects in the namespace are also copied over automatically.

Peter demonstrated a GUI example that showed an animated bouncing duck in a window. He was able to change GUI properties and settings "on the fly." Changes to properties, in the timer object, were immediately reflected in the duck example. All GUI events are now "live" in the workspace, which is a great benefit when you are developing an application and want to test your code and events as you develop.

"Dyadic has pioneered a lot of innovation for the APL developer in this release..."

Another special type of namespace is the session object (`⎕SE`) included with APL/W. Version 7 introduces a Windows-oriented set of menus and toolbar that is visible and remains available when you switch from one workspace to another. A key feature of the session object is that it is fully customizable by the user. This means you can change the structure of the session toolbar by adding, changing or removing menus or menu items. You can even have special sessions for different workspaces including national language support (for example, a French character set). How the session object is configured is fully described in the APL/W manuals. These have been completely rewritten, and are worth noting in themselves.

Peter finished his

(***Namespaces** - continued on page 7)*

*(**Namespaces** - continued from page 6)*
demonstration by showing how easy it is to copy objects to/from the Windows clipboard as well as an example using the native grid object and mortgage tables.

All in all, this was a very impressive demonstration. Dyadic has pioneered a lot of innovation for the APL developer in this release, as in some of its earlier releases. The development of namespaces looks particularly intriguing. All of these features come at a price, however. In version 6.3 you only needed 4MB of RAM. The new memory requirement is now a minimum of 6MB including runtime applications. Increasing hardware requirements should come as no surprise, since new releases of all other languages and development tools usually require more memory and other hardware.

Peter's demonstration was held at **IBM Canada** in the Aetna Tower, TD Centre. The venue was excellent and special thanks to **The Braegan Group** and **Dyadic** for making the arrangements for Peter's visit.

Dyadic is nearly ready to send out version 7.1 [See the announcement on page 8 —Ed]. The next update will improve on VBX support (Visual Basic extensions) and other enhancements. I also think a new modified WDESIGN workspace will be included as well as fixes to outstanding bugs. Future plans for Dyalog APL/W mean getting ready for Windows95 including OLE 2.0, OCX support.

For more information, and to order APL/W, call Diane Boulton or Brian Olsen, **The Braegan Group** in Toronto at (416) 366-6363. Fax (416) 366-4428. See **Dyalog**'s ad on page 9 for the company's contact address in England.

***Robert Byers** works for the **Ontario Ministry of Education and Training** in its Capital and Operating Grants Administration Branch under School Board Operating Grants. The Ministry uses a number of APL interpreters and other software in its day to day work. Robert can be contacted at rbyers@ibm.net or at 73043.1745@compuserve.com.*

APL95NET: An Experiment in Internet Conferencing for APL95

Steve Halasz

A lot of APLers either can't or don't want to spend a lot of money travelling half way around the world to an APL conference, but would very much like to participate in **APL95**.

Concept

The concept is to establish Internet access for on-site attendees and supporting local initiatives for satellite sites to participate remotely.

Pre-conference Activities

- Establish an APL95NET File Transfer Protocol (FTP), Gopher, and World-Wide Web (WWW) site containing conference program, general conference and registration information, and APL95NET instructions, guidelines, and software.
- Recruit assistance in putting together APL95NET, and soliciting and advising local initiatives to establish satellite sites.
- Work out protocols for the use of Internet tools such as email, Internet Relay Chat (IRC), WWW, in conjunction with conference activities.

Conference Activities

- Establish Internet access and training for on-site conference attendees.
- Make conference proceedings, or at least summaries, available on-line on WWW.
- List conference attendees, both on-site and remote, and include email address, and perhaps bios, on WWW.
- Vendor exhibits on WWW and email and IRC protocols for vendors.
- Special on-site APL95NET "reporters" issuing dispatches on the Net describing conference activities in colourful detail.
- Organize use of email and IRC around scheduled conference activities, such as presentations, birds-of-a-feather (BOF) sessions, vendor forums.
- Log email and IRC threads so that participants can browse them via WWW at leisure.
- Organize real-time broadcast over the Net of one presentation (of at least audio and possibly video) with the possibility for remote sites to submit questions via IRC in real time.

Costs

There is no funding, but I feel the resources needed can be "scrounged" on site. It will be up to remote sites to establish their own access to the net to

*(**APL95NET** continued on page 12)*

Dyadic Announces Dyalog APL for Windows Version 7.1

Dyadic is pleased to announce Dyalog APL/W Version 7.1 for Microsoft Windows. This is a maintenance release that will be distributed to customers free of charge. It does however contain a significant number of enhancements.

As further evidence of Dyadic's commitment to provide greater compatibility with IBM APL2, Dyalog APL/W Version 7.1 includes the following language enhancements: Enclose with axes, Take and Drop with Axes, Ravel with Axes, and Strand Assignment with parentheses. These enhancements do not conflict with existing Dyalog APL language conventions and are implemented at all migration levels (defined by the system variable ⎕*ML*). In addition, Version 7.1 offers an optional APL2-compatible partitioned enclose at migration level 3.

The Dyalog APL/W session now supports *drag and drop* editing which is implemented in a manner that is consistent with Microsoft Word for Windows. Drag and drop editing provides a fast and convenient method for moving and copying text (both whole lines and partial lines) within an edit window or between edit and session windows.

Namespaces have been extended in several ways. You may now create a GUI object, such as a Form, as a child of a namespace. Indeed, you can insert a namespace at any level in the GUI hierarchy. For example, you could create a namespace as a child of a Form and then create Buttons and other objects as children of the namespace together with any code and data that the objects need to share. Secondly, namespaces and GUI objects may be stored on component files in their ⎕*OR* form. This feature will greatly simplify the management and reusability of complex objects and provides the basis for the implementation of *class libraries* in the future. Further enhancements to namespaces may also be included in Version 7.1.

The popular Grid object now supports the selection of rows, columns and blocks of cells. A selected block can be cut or copied to the clipboard and pasted back into the Grid. This facility also allows the user to transfer data very quickly between the Grid object and other spreadsheets such as Microsoft Excel. You may also drag a block of cells and drop them elsewhere within the Grid. Individual rows and columns may be resized by the user dragging the row and column title dividers. The user may also have the system resize a row or column to fit the contents of that row or column, by double-clicking the mouse over a divider. All of these operations generate new events to which you may attach callback functions and you may also generate these actions under program control. A new mechanism is provided to control the input mode and the behaviour of the cursor keys. All these features have been implemented in a manner that is consistent with Microsoft Excel.

In addition to these changes, the Grid now allows you to associate Combo and Button objects with individual cells. Combos provide a very convenient means for the user to input one of a series of options and Buttons (particularly Check boxes) provide a good way of making and displaying yes/no choices. As Button objects may display bitmaps, icons and metafiles, you can also use them to display pictures in individual Grid cells.

In addition, Dyalog APL/W Version 7 users will find a large number of minor enhancements. Dyadic intends to ship Version 7.1 during January.
(See advertisement on opposite page.)

***APL* People**
Looking for *APL* work?

⇨⇨⇨ 💾 ⇦⇦⇦
⇩

***APL* Companies**
Looking for *APL* staff?

☑ Check This!

The APL Skills Database

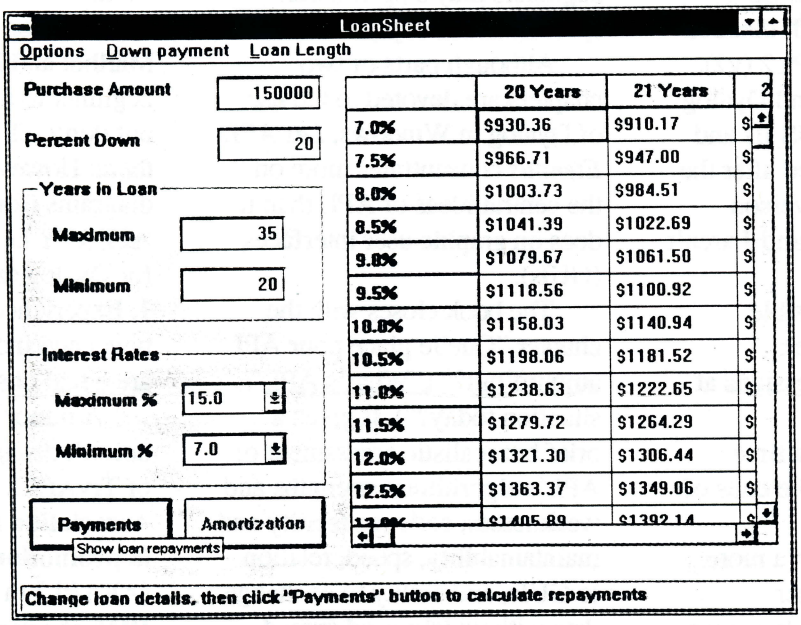
A FREE employment referral service offered by
The Toronto *APL* Special Interest Group

Matching *APL*-Skilled People with Employers who need *APL* Skills
Contact the Toronto *APL* Special Interest Group for more information

dyalog APL

The Definitive APL for Windows™

Thanks to numeric edit fields, automatic hints and tips, and the powerful built-in Grid object, applications like this are a snap in Dyalog APL/W.



Experience counts

Since we launched Dyalog APL/W, nearly two years ahead of our nearest competitor, our customers have developed hundreds of successful industrial-strength GUI applications. With the benefit of their experience, we have enhanced and refined Dyalog APL/W into the mature, stable and above all useful Windows development tool that it is today.

Not only does Dyalog APL/W provide the most comprehensive set of GUI facilities available for any APL, but you can be confident that your workspaces will run unchanged on Unix workstations under OSF/Motif.

With Version 7.0 we have added support for Visual Basic Custom Controls, a powerful built-in Grid object, Numeric, Currency, and Date fields, ToolBar, StatusBar and TabBar objects, automatic context-sensitive Hints and Tips, Metafiles, MDI, 3-D Forms and Controls, a fully customisable Session, an ODBC interface, Namespaces for encapsulation, and a host of other improvements.

That's why Dyalog APL/W remains the professional choice. For a free trial copy, contact Dyadic or your local distributor today.

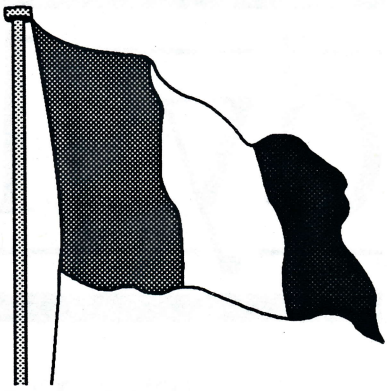
Dyadic Systems Limited., Riverside View, Basing Road, Old Basing, Basingstoke, Hants. RG24 7AL, United Kingdom. Tel:+44 256 811125 Fax: +44 256 811130 Email: sales@dyadic.com.

Microsoft is a registered trademark and Windows and the Windows Logo are trademarks of Microsoft Corporation

MICROSOFT® WINDOWS™ COMPATIBLE

A pleasure to read and a pleasure to learn from...

Dick Holt



This book has two goals: to help beginners to discover APL progressively, and to serve as a reference for more advanced APL users. It meets both goals well. This book is the widest, and deepest, coverage of APL I've seen since *APL An Interactive Approach* (Gilman & Rose) a decade ago.

A hefty nineteen chapters (588 pages) span all of the central concepts of APL and APL2 in their various forms, with detailed comparisons among APL2 (v2), APL*PLUS II (v5.1) and Dyalog APL (v6.3). APL*PLUS III and Dyalog v7 were released after the book was begun, and Iverson Software's APLIWIN and J aren't covered.

Legrand offers sensible guidance to the beginner:

- test your APL expressions at the keyboard
- take it a little bit at a time
- do the exercises at the ends of the chapters
- seek assistance from a more experienced APL user
- skip around; the book is not a novel

From there, the book goes straight to data structures, including mixed (or generalized) arrays and their importance in APL. The next several chapters cover primitives and defined functions, data transformations, operators (each, first, pick, enclose, etc.).

As you would expect, the mechanics of `)SI`, format, execute, ws management, files, auxiliary processors, editing, error trapping, etc. are all discussed in detail, with separate examples for APL2, APL*PLUS, and Dyalog where applicable. Almost every chapter has exercises designed for the beginner (with answers at the back of the book) as well as a "specialist's corner" that beginners may skip and that specialists will enjoy.

Although parts of two chapters are devoted to the use of Dyalog in Windows, *Les APL Etendus* concentrates more on the central ideas of APL than it does on graphic user interfaces (GUIs).

The book closes with the chapter "Quelle place pour APL aujourd'hui?" ("What's APL's situation today?"). It gives a brief but realistic assessment of APL's algorithmic strengths and weaknesses, code readability and maintainability, speed, relation to GUIs, and ability to exchange data with widely-used spreadsheet and database software.

Yes, this book is in French. Thus, it should be well received widely throughout the French-speaking world. Even those whose main language is English can find it useful. After all, much of the English language has been closely related to French ever since the Norman conquest in 1066. It's a pleasure to read, and a pleasure to learn from.

A minor complaint: both Mailhol and Legrand exhort the beginner to use keyboard stickers, but don't tell you where to acquire them. However, keyboard diagrams (for the French AZERTY keyboard) are provided for Dyalog and APL2 in Annexe 1. Experience shows that APL's novel and inconsistent keyboards are a real obstacle for beginners.

A minor puzzlement: Legrand doesn't discuss the French version of TryAPL2 produced by the **Association Francophone pour la promotion du Language APL (AFAPL)** in cooperation with **IBM of France**. This free version of TryAPL2 includes a full-screen editor (not included in the original TryAPL2) that frees the APL learner from the long obsolete del editor.

This French version of TryAPL is available from AFAPL at 160-162 rue Cardinet, 75017 Paris. A knowledge of French isn't needed to use this full-screen editor.

Bernard Legrand (1994)
Les APL Etendus: APL2, APL*PLUS, Dyalog APL sous Windows
Masson, Paris. ISBN: 2-225-84579-4. Preface by B. Mailhol
120 boulevard Saint-Germain, 75280 Paris Cedex 06, France
Via Statuto 2/4, 20121 Milan, Italy
Avenida Principe de Asturias 20, 08102 Barcelona, Spain
SMPF Inc., 100 East 42nd Street, Suite 1001-2, New York, NY 10017, USA.
Tel: +1 (212) 983-6278. Fax: +1 (212) 687-1407. **Price**: FF400.00, $US80.00.

Dick Holt *publishes French software on the APL BBS\APL, has translated French software into English, and has promoted the AFAPL version of TryAPL. His own software (used among fishing cooperatives in Francophone West Africa) has been translated into French. dick.holt@acm.org or*

The Gilman & Rose of the 1990s?

Adrian Smith

First some statistics: this is a substantial and comprehensive textbook of APL comprising over five hundred pages in seventeen chapters; it covers all aspects of the APL language, and makes clear the differences in dialect between APL2 and the other two implementations, e.g., the "*Deux Philosophies*" (page G21) towards expressions like `A B C[2]` where the '*Force de liaison*' differs in APL2.

All the chapters close with a set of exercises. These are nicely chosen to make you think. They avoid excessive repetition or trick questions which trip up unwary readers and are liable to put them off APL altogether. It is nicely presented, with good indexing and a helpful set of summary tables showing (for example) all the possible reductions applied to the empty vector.

Coverage

Chapter A covers a little history, and introduces the various APL keyboards. Chapter B takes you through variables and assignment, simple arrays of various ranks, and vector notation. It makes clear the differences between *mixed* and *general* arrays, and also notes the index primitive ⌷ offered by APL2 and the extension to bracket indexing `MAT[(1 2)(3 4)]` offered by Dyalog APL.

Chapters C and D are a straightforward introduction to the 'core' APL primitives (with a good discussion of reduction and behaviour on empty arrays), and the basics of function definition. Again, the different rules for forming the function header (such as the need to include {} around optional arguments in Dyalog) are well explained. Chapter E continues with a discussion of errors and the State Indicator (although I must take issue with Bernard's definition of `NONCE ERROR` as something specifically related to a failed interface call), and has quite a comprehensive comparison of the various tracing tools available in the different environments.

Chapters F and G are all about restructuring and transforming arrays. In general, Chapter F is 'classical' APL, with the addition of selective specification and the use (in APL2) of an axis modifier for ↑ and ↓. The discussion of dyadic transpose is particularly thorough and supported with several helpful diagrams. Chapter G walks you through the use of the `DISPLAY` utility, and explains very clearly the meaning of the various indications it uses to hint at the type and rank of your data. It is also used to illustrate exceptionally clearly the different effects of *catenation* and *juxtaposition* on the structure of the resulting vector. This is (in my experience) one of the biggest problems for a 'flat' APLer to overcome in moving to an extended system.

Chapter H covers execute and (in some detail) format. Here the various APLs diverge dramatically, so the APL2 'format by example' is given quite a separate treatment to the use of `⎕FMT`. Again, both are well illustrated with examples. Chapter I discusses operators, with some nice examples of inner products. Chapter J sweeps up all the remaining math functions, with examples of ⌸ in curve fitting.

We now move on to the environment, with the various `⎕fns` (such as `⎕RL` and `⎕PW`) outlined in Chapter K and the basics of shared variables (including DDE from Dyalog APL) in Chapter L. Chapter M is a good basic introduction to file processing in all three APLs and Chapter N introduces the event-handling facilities. Unfortunately these differ so completely in the three environments that a sensible treatment is more or less impossible, and I doubt whether the coverage is sufficient to enable readers to program these with total confidence and safety. Finally, Chapter O covers editing, from the ever-present ∇ to the windowed facilities available with Dyalog.

Chapter P is a nice gentle introduction to the Dyalog approach to Windows development (unfortunately the book pre-dates the APL*PLUS III release) with some good examples and lots of screen-shots. Chapter Q has the obvious (but it needs saying) advice on documentation and housekeeping ("*Assurez l'hygiène de votre environnement*") and the book closes with some thoughts on "Why APL Today." Bernard takes many of the accusations levelled at APL and hits back at (for example) unreadability with some well-chosen counter-examples from Excel.

Summary

This is the first really good book I have seen which takes an integrated approach to the extended APLs, rather than

*(**Legrand** continued on page 12)*

*(**APL95NET** continued from page 7)*
participate. I'm counting on the fact that a lot of people want very much to see the use of the Internet expanded to new dimensions and will do what they can to put this initiative together.

Off-Site Registration

Off-site participants will be asked to register but a registration fee will not be charged on the grounds that, for the first year, off-site access will be limited and experimental.

Results

It should be possible to increase conference participation by from two to five times as many people. Many people will be able to participate in a limited way who would not attend the conference site. For on-site attendees, there will be a whole new dimension to attending the conference. This will include Internet services available to them while at the conference and the additional interaction available with remote sites. Both of these possibilities should make the conference much more interesting and fulfilling for on-site attendees.

Future

I believe that APL95NET will very much enhance the conference immediately. My real hope is that in a few years—as we develop experience with it and as the Internet itself "grows up"—we can decentralize the conference into numerous "peer" sites using Internet teleconferencing. As a result, there could be thousands of participants at future conferences.

*Steve **Halasz** has been an APL database and financial applications developer since 1980 and works as an APL consultant with **Lingo Allegro U.S.A., Inc.** in Chicago, Illinois. He can be contacted at SJHalasz@InterTrieve.com.*

*(**Legrand** continued from page 11)*
grafting general arrays on to the back of an existing text. It deserves to be the Gilman & Rose of the 1990s. Someone should translate it, but please let that someone be an APLer with a smattering of French, not a professional translator who knows no APL!

***Adrian Smith**, producer of the British APL Association's quarterly journal **Vector**, and creator of the graphical user interface product, CAUSEWAY (reviewed in **Gimme Arrays!**'s September issue) can be contacted at +44 439 788385 or 100331.644@compuserve.com.*

*We wish to thank **Vector** for permission to reprint this article from its October 1994 issue. **Gimme Arrays!** readers who have been following Clifford Reiter's work since he presented to us here in Toronto last year will be pleased to see a lengthy article by him in the same issue of **Vector**.*

Calendar of Events

Local

February 8, 1995

Achieving usable documentation. Stephanie Rosenbaum, Tech-Ed. Full-day workshop sponsored by the ITRC and the Toronto Chapters of ACM SIGCHI (ToRCHI) and STC. Time: 9:00am -5:00pm. Location: Sutton Place Hotel. Cost: $101.65 for members of the above groups, otherwise $203.30. To register before Feb. 1, telephone (416) 978-8558; email: rosanna@itrc.on.ca.

March 8, 1995

Art and technology: Interface issues. Derrick de Kerckhove, Faculty of Information Studies and the McLuhan Centre for Research. Sponsored by the Toronto Regional chapter of ACM SIGCHI (ToRCHI). Time: 7:00pm - 9:30pm. Location: Room 307, Faculty of Information Studies, University of Toronto, 140 St. George St., Toronto. Free to ToRCHI members, otherwise $3 at the door.

April 1, 1995

Communicating by design. Professional development day with workshops and luncheon awards presentation. Sponsored by the Society for Technical Communication (STC). Cost: $95 for STC members, otherwise $120. Contact: Fax (416) 784-6370, Attention Stephanie Copp.

July 16-25, 1995

The International Mathematical Olympiad. For high school mathematics students from seventy-five countries. Location: York University, North York, Ontario. Contact: **Richard Brown** at Richard.Brown@mathstat.yorku.ca. Phone (416) 736-5250. Fax (416) 736-5757.

International

June 4-8, 1995

APL95: APL Does It Today. San Antonio, Texas. Sponsored by ACM SIGAPL. Contact: APL95_info@acm.org, APL95_conf@acm.org, or APL95, PO Box 210367, Bedford, TX 76095 USA. See page 5 for important dates.